REMARKS

Correspondence address change

Applicant notes that a correspondence address change form has been submitted herewith, and requests that the correspondence address for the current patent application be correspondingly updated.

Claim rejections under 35 USC 102(e)

Claims 1, 6-7, 10-11, 13-14, 17, and 20 have been rejected under 35 USC 102(e) as being anticipated by McIntosh-Smith (6,324,632). Claims 1, 11, and 17 are independent claims, from which the remaining pending claims depend. Applicant asserts that claims 1 and 17, as amended, and claim 11, as originally filed, are patentable over McIntosh-Smith. For at least the same reasons, the remaining pending claims are also patentable over McIntosh-Smith.

Claims 1, 11, and 17 are limited to the memory line location-dependent cache allocation policy in particular "*specifying which lines of memory are to be allocated at which of the plurality of cache sections based on the locations of the lines of memory*." As an example, the allocation policy may specify that data from memory lines at locations 100-200 are to be allocated to a first cache section, and data from memory lines at locations 200-300 are to be allocated to a second cache section. That is, the allocation policy specifies cache sections for memory lines based on the locations of the memory lines.

Applicant submits that McIntosh-Smith does not disclose such an allocation policy. The Examiner indicates that either the vector max/min algorithm of FIG. 4 of McIntosh-Smith, or the random generator number algorithm of FIG. 6 of McIntosh-Smith, discloses a memory line location-dependent cache allocation policy. Applicant disagrees. As a general note, Applicant comments that neither the algorithm of FIG. 4 nor the algorithm of FIG. 6 disclose anything comparable to a "memory line location" in determining at which cache section to cache a given

memory line. However, the cache allocation policy of claims 1, 11, and 17 is specifically limited to such a policy, in which the memory line location determines at which cache section a given memory line is cached. Applicant now specifically discusses these algorithms individually.

With respect to the vector max/min algorithm of FIG. 4 of McIntosh-Smith, it is an algorithm for "executing a vector maximum algorithm using the lines of code illustrated in Table 2." (Col. 6, ll. 3-5) As apparent from the code of Table 2, the algorithm looks at the vectors vector[0] through vector[vectorsize], and sets the variable max equal to the largest of these vectors. (Col. 6, ll. 39-45) However, determining the largest vector within a group of vectors does not having anything to do with a policy that specifies which lines of memory are to be allocated at which cache sections based on the locations of the lines of memory, as to which claims 1, 11, and 17 are limited.

More particularly, the vector maximum algorithm of FIG. 4 is described as using a cache only in line 5 of Table 2 (col. 6, l. 36), in which the algorithm creates a translation table in a cache and sets "cache behavior bits . . . to indicate that writeback mode should be used when accessing data falling within this translation." (Col. 6, ll. 64-67) However, absent from this description is that memory lines are allocated to cache sections *based on the locations of the memory lines*, as to which claims 1, 11, and 17 are limited. The setting of cache behavior bits indicates some type of special writeback mode is used when accessing data falling within the given translation. However, whether data falls within a given "translation" does not equate to *the locations of memory lines* being examined in determining at which cache sections those memory lines should be cached; the algorithm does not have to look at memory line locations to see if data falls within the translation, and, more significantly, is not disclosed in McIntosh-Smith as looking at memory line locations. Therefore, the vector algorithm of FIG. 4 of McIntosh-Smith does not anticipate claims 1, 11, and 17.

With respect to the random number generator of FIG. 6 of McIntosh-Smith, it is "a process executing a random number generator," where "Table 3 lists the code for this example."

(Col. 7, ll. 20-22) That is, the code of Table 3 generates a pseudo-random number. Generating a random number, however, does not have anything to do with a policy that specifies which lines of memory are to be allocated at which cache sections based on the locations of the lines of memory, as to which claims 1, 11, and 17 are limited.

More particularly, the random number generator process uses a cache as follows:

> As the random number generator proper begins, memory references to the "output" array occur, causing cache misses and subsequent cache line refills. In a cache without partitioning, a large "output" array could easily start to evict the code for the "pseudorandom" function itself from the cache. Cache partitioning avoids these problems of stream data evicting closed end program data from the cache.

(Col. 7, ll. 40-46) Notably absent in this utilization of a cache by the random number generator process is determining which memory lines should be cached at which cache sections *based on locations of the memory lines*. Indeed, the random number generator process looks at the *type* of data being cached, *not* its location in memory. For instance, it appears that "stream data" is placed in one section or partition of the cache, and "closed end program data" is placed in another section or partition of the cache. However, McIntosh-Smith does not accomplish such cache partitioning by looking at *locations of memory lines*, as to which the claimed invention is limited, but rather looks only at the *type* of data, either stream data or program data. Therefore, the random number generator process of FIG. 6 also does not anticipate claims 1, 11, and 17.

Claim rejections under 35 USC 103(a)

Claims 2-5, 8-9, 12, 15-16, and 18-19 have been rejected under 35 USC 103(a) as being unpatentable over McIntosh-Smith in view of Douglas (5,537,635). Applicant notes first that all of these claims are dependent claims depending from independent claim 1, 11, or 17, and therefore they are patentable for at least the same reason that independent claims 1, 11, and 17 are. Furthermore, Applicant submits that amended claims 5, 10, and 19 and originally filed claim 12 are independently patentable, irrespective of the patentability of claims 1, 11, and 17. That is,

Applicant submits that claims 5, 10, 12, and 19 are patentable over McIntosh-Smith in view of Douglas. Although the Examiner rejected claim 10 under 35 USC 102(e) as being anticipated by McIntosh-Smith, Applicant has amended claim 10 such that it recites more comparable limitations to those of claims 5, 12, and 19. Therefore, Applicant has grouped claim 10 together with claims 5, 12, and 19, and discusses its patentability along with claims 5, 12, and 19 over McIntosh-Smith in view of Douglas.

Claims 5, 10, 12, and 19 are limited to dynamically adjusting the cache allocation policy based on the temporal locality of memory line locations, and/or specifying that one cache section is for caching memory lines that have one type of temporal locality and another cache section is for caching memory lines that have another type of temporal locality. Applicant believes that there may be some confusion as to what "temporal locality" is. Temporal locality is how close, or local, the memory is to a processor with respect to time. For instance, a processor may have high temporal locality memory, communication with which takes a long time to get to the processor, and low temporal locality memory, communication with which takes a short time to get to the processor. The low temporal locality memory may be located in the same computer as the processor, for instance, and its communication gets to the processor in a relatively short period of time. The high temporal memory may be located outside of the processor's computer, across a network, and therefore its communication gets to the processor in a relatively long period of time.

Now, this is the important part: *the actual performance type of the memory is not determinative of the temporal locality of the memory.* For instance, in the example in the previous paragraph, the low temporal locality may be of high latency, such that it is relatively poor performance memory, whereas the high temporal locality may be of low latency, such that it is relatively high performance memory. However, the low temporal locality of the high latency memory owes more to its *location relative to the processor in time (i.e., it is temporally close to the processor)*, than it does to the relatively poor performance nature of the memory. Similarly, the high temporal locality of the low latency memory owes more to its location relative to the

processor in time *(i.e., it is temporally far away from the processor)*, than it does to the relatively high performance nature of the memory. That is, the *type* of a given memory is *not determinative* of the *temporal locality* of the memory. The memory could be super-duper fast memory with super-duper low latency, but if it is located on a slow network away from a processor, it is high temporal locality memory regardless. Similarly, the memory could be very slow memory with incredibly high latency, but if it is located right next to the processor, it can be low temporal locality memory as compared to the memory located on a slow network away from the processor.

Against this background, Applicant submits that it is clear that McIntosh-Smith in view of Douglas does not render claims 5, 10, 12, and 19 obvious. In particular, the Examiner relies upon column 4, lines 16-27, and column 1, line 61 through column 2, line 13 of Douglas in disclosing dynamically adjusting the cache allocation policy based on temporal locality, and/or specifying that one cache section is for caching memory lines with one type of temporal locality and another cache section is for caching memory lines that have another type of temporal locality. Applicant discusses each of these excerpts in turn, but notes generally that neither excerpt of Douglas discloses *temporal locality of memory*, as has been described above.

Column 4, lines 16-27 of Douglas read as follows:

> The method continues by specifying reclaim vectors necessary to achieve the target sizes. A reclaim vector is assigned to each partition and contains a list of numbers. Each number indicates how many frames should be stolen from a corresponding partition. The total number of frames specified by the reclaim vector is equal to the amount necessary to achieve that partition's target size.
>
> As an example of how a reclaim vector is set, assume a cache has four partitions and that the optimization algorithm determines that partition 1 should contain 100 more frames than it currently does. Furthermore, an assignment algorithm determines that 50 of these frames should come from partition 2 and 50 should come from partition 3.

This discussion of dynamically adjusting cache allocation in Douglas does not discuss or disclose temporal locality at all, and further does not discuss the idea of allocating low temporal locality memory lines to certain cache sections and high temporal locality memory lines to other cache

sections. Column 4, lines 16-27 of Douglas do talk about dynamically adjusting cache allocation, but they specifically talk about *how* to adjust the cache allocation, not *on what basis* (such as temporal locality) the cache allocation should be adjusted – and claims 5, 10, 12, and 19 are particularly limited to one special type of basis, temporal locality. Therefore, column 4, lines 16-27 of Douglas do not provide a basis for McIntosh-Smith in view of Douglas rendering claims 5, 10, 12, and 19 obvious.

Next, column 1, line 61 through column 2, line 13 of Douglas read as follows:

> A partitioned cache is one whose storage is divided into separate domains or partitions. *Data with similar performance requirements are put into the same partition. The reason for this is to isolate the effects of managing high performance data from low performance data and vice versa.* This allows a greater range of data to be effectively managed by a single cache. It may be implemented as a software cache or as a hardware cache, for example in the disk drive controller. A partitioned cache is of a fixed size, but the size of the partitions that make up the cache may vary through a process of dynamic reallocation of memory. These dynamic caches are controlled by a program operating on a processor or controller. Such caches employ some method of determining optimal partitions sizes and shifting resources between the partitions to achieved the optimum size as it is updated. *For example, this optimization may be based on the hit/miss ratio performance of each cache.* Other partitioned caches statically allocate resources to each partition and avoid the problems inherent in dynamic reassignment of cache resources.

(Emphasis added) This discussion does not disclose two aspects of claims 5, 10, 12, and 19: (1) *temporal locality* of (2) *memory line locations*. First, with respect to temporal locality, column 1, line 61 through column 2, line 13 of Douglas does not disclose temporal locality, but rather talks about "high performance data" and "low performance data." However, as has been described above, high performance data can still have high temporal locality – and end up being relatively slow – because it is far away from the processor, and low performance data can still have low temporal locality – and end up being relatively fast – because it is close to the processor. However, claims 5, 10, 12, and 19 are specifically limited to temporal locality. Therefore, column

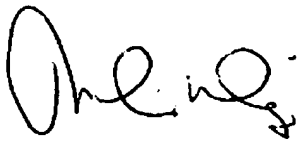
1, line 61 through column 2, line 13 of Douglas does not provide a basis for McIntosh-Smith in view of Douglas rendering claims 5, 10, 12, and 19 obvious.

With respect to memory lines, column 1, line 61 through column 2, line 13 of Douglas does not disclose "memory line locations" as being the determinative factor of cache allocation, but rather "data." This is an important distinction. Memory line locations can store data, but given data can be stored at many different memory line locations. A location of memory is not the same as the data it stores. Douglas is concerned about cache allocation based on the performance requirements of the *data* that can be stored in memory line locations – not of the temporal locality of *where* this data is stored, which are the *memory line locations* at which the data is stored. However, claims 5, 10, 12, and 19 are specifically limited to *memory line locations*, not data, as being determinative of cache allocation. Therefore, column 1, line 61 through column 2, line 13 of Douglas does not provide a basis for McIntosh-Smith in view of Douglas rendering claims 5, 10, 12, and 19 obvious.

Conclusion

Applicants have made a diligent effort to place the pending claims in condition for allowance, and request that they so be allowed. However, should there remain unresolved issues that require adverse action, it is respectfully requested that the Examiner telephone Applicants' Attorney so that such issues may be resolved as expeditiously as possible. For these reasons, this application is now considered to be in condition for allowance and such action is earnestly solicited.

Respectfully Submitted,



6-25-2004
Date

Michael A. Dryja, Reg. No. 39,662
Attorney/Agent for Applicant(s)

Law Offices of Michael Dryja
704 228th Ave NE #694
Sammamish, WA 98074
tel: 425-427-5094
fax: 206-374-2819